

FIG. 1

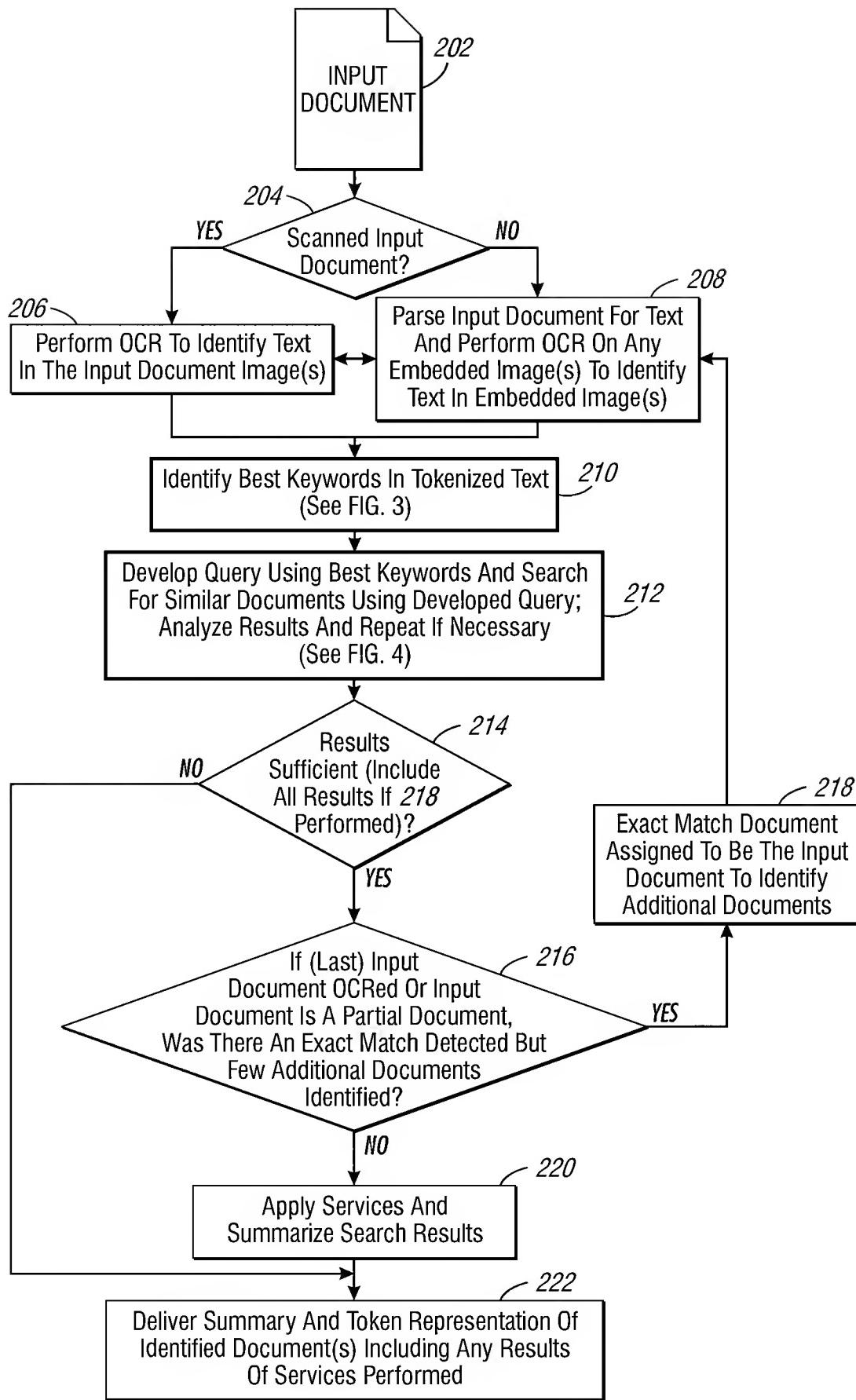


FIG. 2

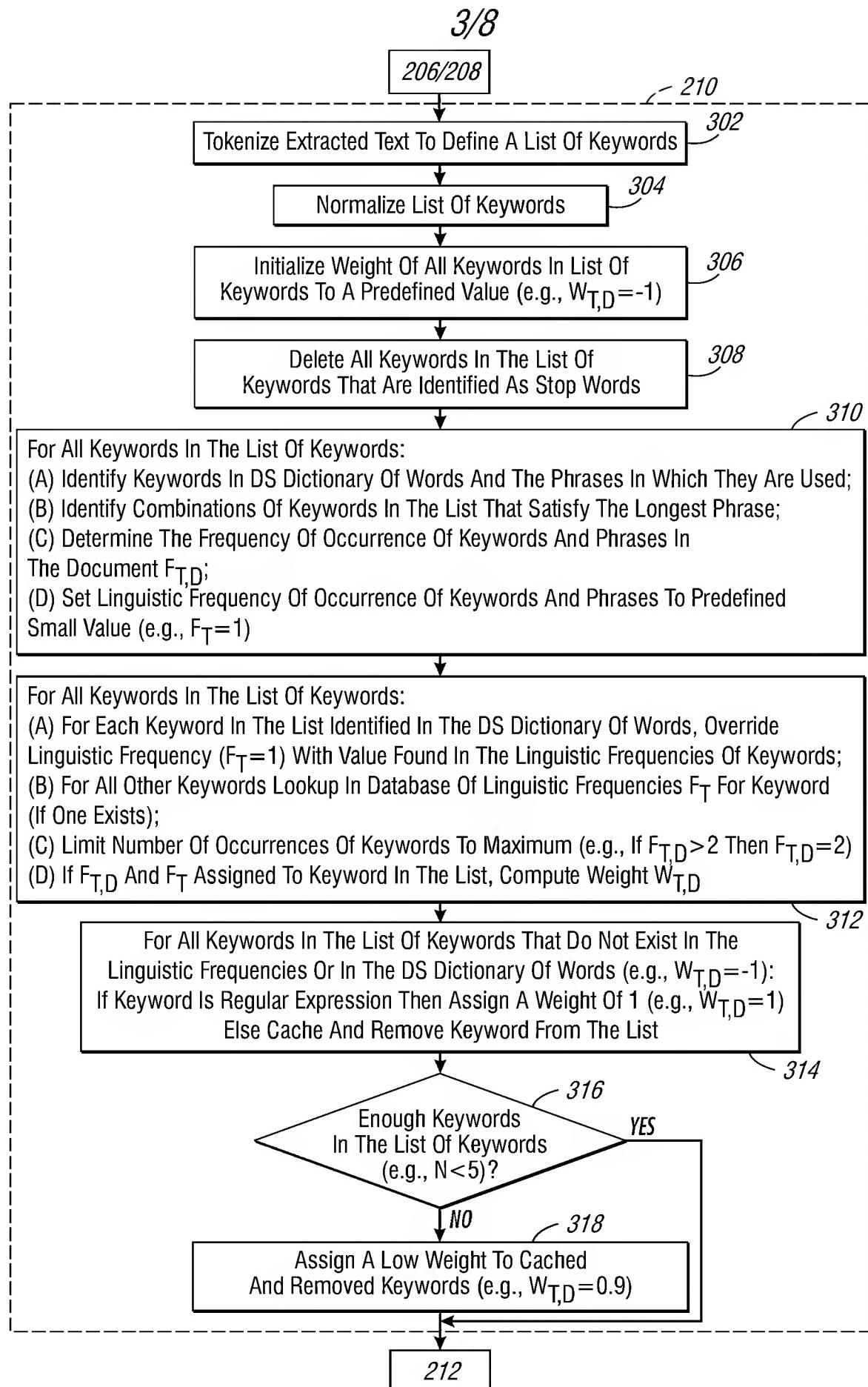


FIG. 3

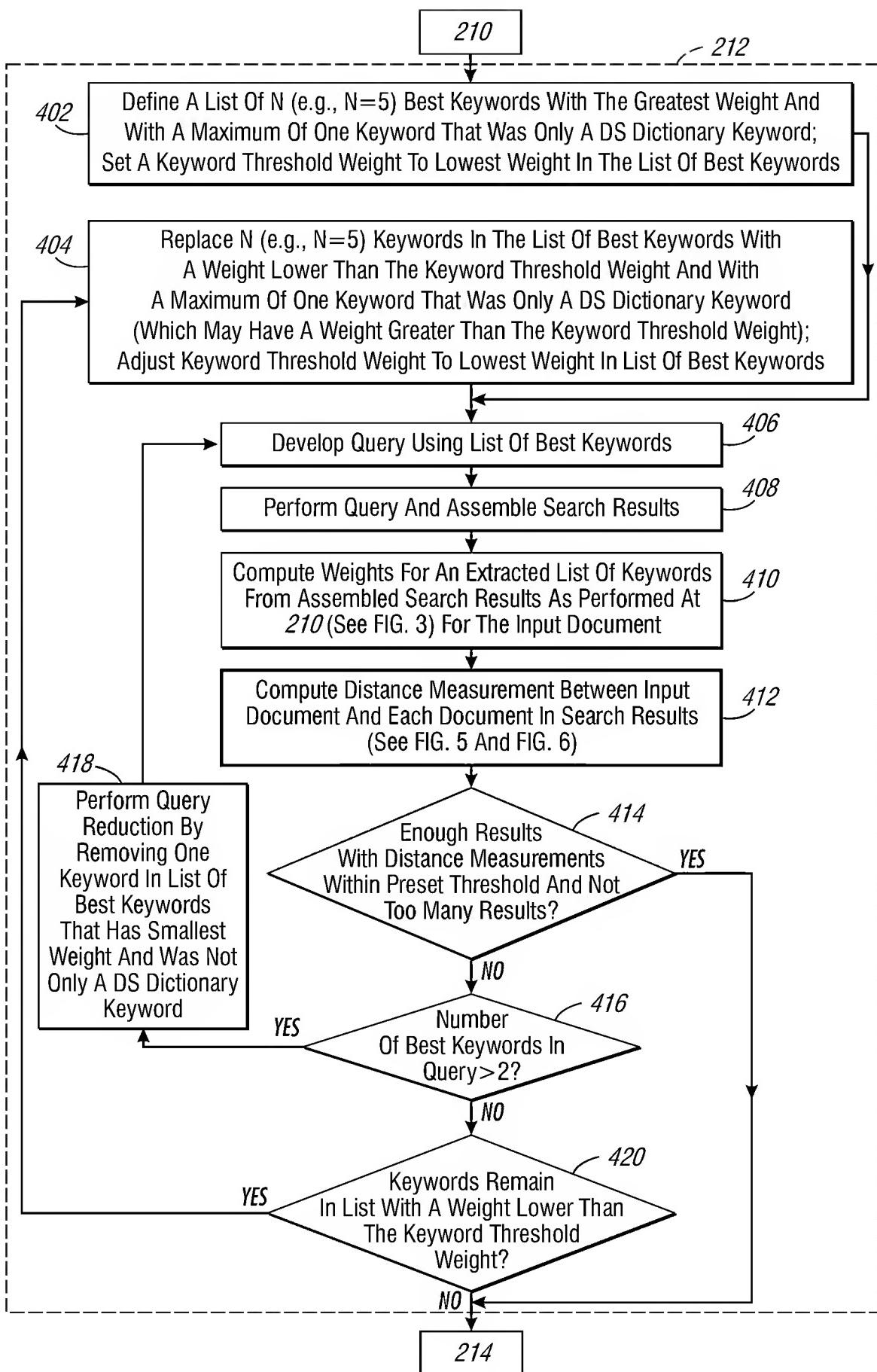


FIG. 4

CALCULATE_SIMILARITY [D1,D2] -calculates the similarity between documents D1 and D2

- Input:
 - D1: List of keywords of the input document
 - D2: List of keywords of document from search results
- Output:
 - Similarity “S”: The computed similarity between D1 and D2

502 { For the document D1, calculate with the keyword weights of D1:

- unique attributes sum: Sum1 = the sum of the weights of keywords in D1 that do not appear in D2
- total sum: Sum2 = the sum of the weights of keywords in D1
- shared sum: Sum3 = the sum of the weights of keywords in D1 that also appear in D2
- ratio: R = (the number of keywords in D1 not in D2)/(the number of keywords in D1)

}

504 { For the document D2, calculate with the keyword weights of D2:

- unique attributes sum: Sum4 = the sum of the weights of keywords in D2 that do not appear in D1
- shared sum: Sum5 = the sum of weights of keywords in D2 that also appear in D1

}

506 { If D1 originates from a hardcopy document, calculate the tolerance ratio “T”:
 K = a constant defined by OCR error rate at the keyword level, if no OCR error is detected, K is set to 0
 $T = K * (Sum2 - Sum3) / (Sum2)$

508 { Calculate the inclusion ratio “I” (i.e., percentage of keywords from D1 that are in D2):
 $I = (Sum3) / (Sum2) + T$

TO FIG. 6

FIG. 5

If $I>90\%$ (i.e., if an inclusion is detected, e.g., 90 % of the keywords from D1 are in D2):
 $\text{Sum6} = \text{Ordered_Sum}[D1, D2]$ - sum of the weights of keywords in D1
 with same neighbors in D2 (SEE FIG. 7)

- Calculate ordered inclusion ratio “ $I2$ ”:

$$I2 = (\text{Sum6}) / (\text{Sum2})$$

if ($I2 > I$) then $S = I$

else

if ($D1$ originates from a hardcopy document and $I2 > 50\%$)
 if ($R < 20\%$) then $S = I$ else $S = I2$

else $S = I2$

else (i.e., if no inclusion is detected):

- Calculate the Jaccard similarity distance measure:

$$\text{Sum7} = \text{Sum1} + \text{Sum4} + \text{Sum5}$$

$$S = (\text{Sum5}) / (\text{Sum7})$$

- If $S > 90\% :$ (a revision is detected, i.e., Jaccard similarity $S > 90\%$;
 otherwise a related document may be detected)

- $\text{Sum8} = \text{Ordered_Sum}[D2, D1]$ - sum of the weights of keywords from D2
 with same neighbors in D1

- Calculate ordered similarity:

$$S2 = (\text{Sum8}) / (\text{Sum1} + \text{Sum4} + \text{Sum8})$$

if ($S2 > S$)

if ($D1$ originates from a hardcopy document)

if ($S2 > 50\%$)

if ($R > 20\%$) then $S = S1$

else $S = S1$

else $S = S1$

else $S = S1$

}

510

512

Ordered_Sum [L1, L2] - calculates sum of the weights of keywords from L1 with same neighbors in L2

- Input:
 - L1: List 1
 - L2: List 2
 - Output
 - Sum: the ordered sum

600 Define the tolerance “T”, minimal percentage for neighbors :
 $T = K * 50\%$, where K depends on the OCR error at the keyword level

- for each term t (i.e., keyword) of $L1$:
 - identify all possible positions P_i of term t in $L2$
 - if (t exists in $L2$)
 - identify N neighbors on both sides of term t in $L1$
 - (by default $N=5$ and depends on the position of the term t in the $L1$)

for each position P_j of term t found in L_2 :

602 { _____ if the position P_i of the term t is at a limit of L2: ($P_i < N$ or $P_i > (L2 \text{ size}-N)$)
 increase the ordered sum with the weight of term t in L1: Sum += W_t
 else

```

606 { identify N neighbors on both sides of term t in L2
      Calculate the percentage of common neighbors "C" of term t between L1 and L2
      if ( C > 80% - T)   increase the ordered sum with the weight of term t in L1: Sum += Wt
      else
604 {   increase the ordered sum with the weight of term t in L1: Sum += Wt
}

```

FIG. 7

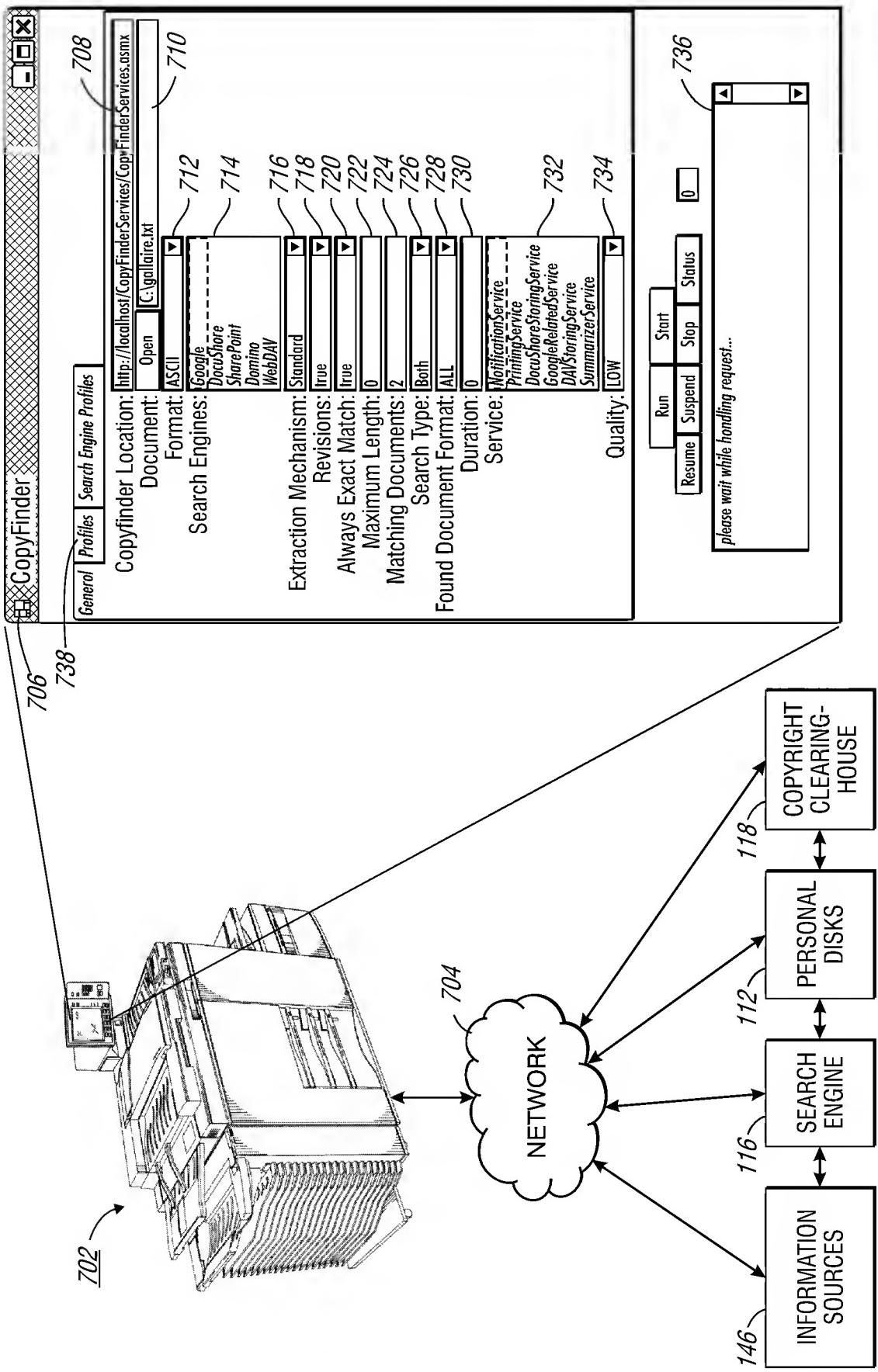


FIG. 8